500.42885X00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants: T. TANAKA, et al

Serial No.: 10/606,955

Filing Date: June 27, 2003

For: PROGRAM GENERATION METHOD

OCT 30 2003 JC141 PATENT & TRADEMARK OFFICE

LETTER CLAIMING RIGHT OF PRIORITY

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

October 30, 2003

Sir:

Under the provisions of 35 USC 119 and 37 CFR 1.55, applicants hereby claim the right of priority based on:

Japanese Application No. 2002-188938
Filed: June 28, 2002

A Certified copy of said application document is attached hereto.

Respectfully submitted,

Carl I. Brundidge
Registration No. 29,621
ANTONELLI, TERRY, STOUT & KRAUS, LLP

CIB/jdc
Enclosures
703/312-6600

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００２年　６月２８日

出願番号
Application Number: 特願２００２－１８８９３８

[ST. 10/C]: [ＪＰ２００２－１８８９３８]

出願人
Applicant(s): 株式会社日立製作所

２００３年　７月　９日

特許庁長官
Commissioner,
Japan Patent Office 太田信一郎

出証番号　出証特２００３－３０５４３５４

【書類名】　特許願

【整理番号】　K02008741A

【あて先】　特許庁長官殿

【国際特許分類】　G06F 11/28

【発明者】

【住所又は居所】　神奈川県横浜市戸塚区戸塚町５０３０番地　株式会社日立製作所　ソフトウェア事業部内

【氏名】　田中　匠

【発明者】

【住所又は居所】　神奈川県横浜市戸塚区戸塚町５０３０番地　株式会社日立製作所　ソフトウェア事業部内

【氏名】　森崎　謙一

【特許出願人】

【識別番号】　000005108

【氏名又は名称】　株式会社日立製作所

【代理人】

【識別番号】　100075096

【弁理士】

【氏名又は名称】　作田　康夫

【手数料の表示】

【予納台帳番号】　013088

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　プログラム生成方法及びその実行処理方法並びにその実施装置

【特許請求の範囲】

【請求項１】

画面を表示もしくは表示制御するための言語で記述された画面ファイルを入力し、該画面ファイルに基づいて、該画面ファイルからデータ取得するパラメータを入力として所定の業務を実行するための業務プログラムの雛型を生成することを特徴とするプログラム生成方法。

【請求項２】

上記画面ファイルを作成する際、該画面フォームでデータ取得するデータの型または属性を定義するステップと、上記業務プログラムを呼出す関数名を指定するステップと、上記業務プログラムが記述されたプログラム言語を指定するステップを有することを特徴とする請求項1記載のプログラム生成方法。

【請求項３】

請求項２記載のプログラム生成方法において、

上記業務プログラムを呼出す際に、予め決められたフォーマットのデータ群から上記業務プログラムを呼出すために必要なデータを抽出するステップと、上記業務プログラムが記述されたプログラム言語のデータ定義に変換するステップとを有することを特徴とするプログラム実行方法。

【請求項４】

画面を表示もしくは表示制御するための言語で記述された画面ファイルを入力する手段と、該画面ファイルに基づいて、該画面ファイルからデータ取得するパラメータを入力として所定の業務を実行するための業務プログラムの雛型を生成する手段とを備えたことを特徴とするプログラム生成装置。

【請求項５】

上記画面ファイルを作成する際、該画面フォームでデータ取得するデータの型または属性を定義する手段と、上記業務プログラムを呼出す関数名を指定する手段と、上記業務プログラムが記述されたプログラム言語を指定する手段を備えたことを特徴とする請求項1記載のプログラム生成装置。

【請求項６】

請求項２記載のプログラム生成方法において、

上記業務プログラムを呼出す際に、予め決められたフォーマットのデータ群から上記業務プログラムを呼出すために必要なデータを抽出する手段と、上記業務プログラムが記述されたプログラム言語のデータ定義に変換する手段とを備えたことを特徴とするプログラム実行装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明はブラウザを通じた画面入力から、画面に連動したプログラムを生成または実行するためのプログラム生成技術やプログラム実行処理技術に関する。

【０００２】

【従来の技術】

画面表示言語で記述されたファイルから取得した入力情報をサーバ側で処理するシステム開発において、入力パラメータは、全て統一フォーマットの一括送信という形が広く普及している。その為、パラメータを分割し、必要なデータを取り出す処理をテンプレート化して、画面情報を元に、業務アプリケーションのインタフェースを作成する手法が知られている。

【０００３】

【発明が解決しようとする課題】

上記パラメータを分割するところまでは可能であるが、分割された個々のデータを様々な言語でコーディングされた業務アプリケーションと連携させるためには、各言語特有のノウハウが必要となる。画面を表示するプログラムと親和性のある言語の場合はあまり問題ではないが、画面を表示する機能が少ないもしくは機能を有さないプログラム言語（COBOL等のような）で記述されたプログラムと連携する場合、インタフェースが作成されても、それを利用した場合、プログラムの開発効率や再利用性が低下してしまうという課題があった。

【０００４】

本発明の目的は、上述の従来型における課題に鑑み、プログラム開発者が所望

するプログラム言語でコーディングされた業務プログラムを容易に連携できるように上記プログラム言語でコーディングされた業務プログラムの雛型を作成することを可能とするプログラム生成方法およびその実行処理方法並びにその実施装置を提供することにある。

【０００５】

【課題を解決するための手段】

上記目的を達成するため、請求項１が関わる発明は、画面ファイルを作成することで業務プログラムの雛型を自動生成することができるプログラム生成方法を特徴とする。

【０００６】

請求項２が関わる発明は、請求項１で述べたプログラム生成方法を実装するために、画面ファイル上で、入力されるデータを定義するステップと呼出す関数名を定義するステップと開発言語を指定するステップを備えることを特徴とする。

【０００７】

請求項３が関わる発明は、連結されたパラメータから必要なデータを抽出するステップと、それぞれの言語にそったデータ定義に変換するステップを備えたことを特徴とする。

【０００８】

【発明の実施の形態】

以下、図面を用いて、本発明の実施の形態を説明する。

図１にプログラム呼出し部及び生成されたプログラムの処理の実施例を示す。端末１００上で、ユーザからの入力、または業務プログラムからのデータの参照など、データの入出力を必要とする画面が画面Ａ１１０、画面Ｂ１２０、画面Ｃ１３０と遷移する。ネットワーク１４０を介して、それぞれの画面が、Webサーバ１５０にアクセスする。画面からのデータはリクエストという形で、送信される。リクエストの中身は、ユーザからの入力データのほかに、データ送信の際に必要となる様々なデータが、連結された状態になっている。Servlet１６０は画面からの入力データをそれぞれに対応したプログラム呼出し部に転送する役割を持つ。Nativeインタフェース１７０及び１８０はCOBOLまたはJava（JavaはＳｕｎ　Ｍ

ｉｃｒｏｓｙｓｔｅｍｓ，Ｉｎｃ．の商標もしくは登録商標である）に対応したインタフェースであり、データの設定や取得の実装の言語依存部分を隠蔽した機能を持つ。

【０００９】

画面Ａ１１０のアクセスをServlet１６０が受け取り、リクエストを画面Ａ１１０に対応したプログラム呼出部Ａ１１１に渡す。プログラム呼出部Ａ１１１はリクエストの中から、COBOL業務プログラムＡ１１２に渡す必要のあるデータを抽出する。抽出されたデータはCOBOL業務プログラムＡ１１２のパラメータとして指定されているデータ型に変換される。プログラム呼出し部Ａ１１１は変換の終わったデータをNativeインタフェース（COBOL）１７０を介してCOBOL業務プログラムＡ１１２に設定する。最後に、COBOL業務プログラムＡ１１２を実行させる。

【００１０】

画面Ｂ１２０のアクセスはServlet１６０が受け取り、リクエストを画面Ｂ１２０に対応したプログラム呼出部Ｂ１２１に渡す。プログラム呼出部Ｂ１２１はリクエストの中から、COBOL業務プログラムＢ１２２に渡す必要のあるデータを抽出する。抽出されたデータはCOBOL業務プログラムＢ１２２のパラメータとして指定されているデータ型に変換される。プログラム呼出し部Ｂ１２１は変換の終わったデータをNativeインタフェース（COBOL）１７０を介してCOBOL業務プログラムＡ１２２に設定する。最後に、COBOL業務プログラムＢ１２２を実行させる。

【００１１】

画面Ｃ１３０のアクセスをServlet１６０が受け取り、リクエストを画面Ｃ１３０に対応したプログラム呼出部Ｃ１３１に渡す。プログラム呼出部Ｃ１３１はリクエストの中から、Java業務プログラムＣ１３２に渡す必要のあるデータを抽出する。抽出されたデータはJava業務プログラムＣ１３２のパラメータとして指定されているデータ型に変換される。プログラム呼出し部Ｃ１３１は変換の終わったデータをNativeインタフェース（Java）１８０を介してJava業務プログラムＣ１３２に設定する。最後に、Java業務プログラムＣ１３２を実行させる。

【００１２】

それぞれの業務プログラムは必要に応じてデータベース１９０にアクセスを行

う。

図２に本発明を適用したプログラム生成手順を示す。画面ファイル２００内に業務プログラムを作成できるように、業務プログラムの関数名、言語、パラメータのデータ定義、また業務プログラムを呼出すための呼出し部につける名前等の情報を定義情報２０１に定義する。定義情報２０１を生成ツール２１０が読み込んで画面から業務プログラムを呼出すために必要なファイルを生成する。生成物それぞれの役割を説明する。動的データ参照画面ファイル２２０は、画面からの入力や、業務プログラムからデータを取得して表示させるといった、動的にデータの入出力を行う機能を持つ。プログラム呼出し部のソースファイル２３０及びNativeインタフェースのソースファイル２４０は画面と業務プログラムのデータの入出力を仲介をする役割を持ち、データ型の変換やデータ整形、パラメータの抽出等の機能を持つ。業務プログラムの雛型２５０は定義された関数名、画面と連携する部分のデータ定義情報をあらかじめ設定しているので、開発者は、処理の実装部分だけコーディングすればよい。

【００１３】

生成ツールで生成されたファイルを、それぞれマシンに依存したコンパイラ２６０を用いてコンパイルする。それぞれコンパイルしたものが、プログラム呼出し部２７０、Nativeインタフェース２８０、業務プログラム２９０になる。

【００１４】

図３以降に図２に示した生成手順の中に出てきたファイルやプログラムについて、実施例を示しながら説明していく。

図３に画面ファイルの表示例を示す。この画面３００は、名前を入力させるためのテキストフィールド３１０、パスワードを入力させるためのテキストフィールド３２０、フォームデータを送信するための送信ボタン３３０が配置されている。

【００１５】

図４に画面ファイル２００の詳細コードを示す。本例は画面を表示制御するタグ言語で記述されている。定義情報２０１で、jfdという独自の拡張タグを用いて、本発明独自の情報を定義している。下線部２１２で関数情報を定義する。cla

ssName属性にプログラム呼出し部の名称を指定している。dllName属性、programName属性には呼出す業務プログラムのDLLファイル名と実際のプログラム名（関数名）を指定している。Word属性で言語を指定している。scope属性、id属性は動的データ参照画面ファイルのライフサイクルの範囲指定と、プログラム呼出し部にアクセスするための識別子を定義している。下線部２１３でデータ定義情報を定義する。name属性で指定する名称を画面ファイル内の入出力オブジェクトの１つと同一名称にすることで、オブジェクトとの関連付けが行える。interface属性では、業務プログラム内で使用するデータの名前を指定する。size属性ではデータ長を指定する。type属性ではデータ型を指定する。alignment属性にはけた寄せを指定する。paddingChar属性には埋め字の埋める文字（スペース、ゼロ、NULL文字等）を指定する。

【００１６】

図５は生成の段階で、拡張タグによって定義された情報を抜き出してテーブルにしたものである。定義情報５００内の関数情報５０１には画面に対応したプログラム呼出し部の名称がSample.Pageであり、呼出す業務プログラムのDLL名称がSAMPLE、プログラム名がNEWUSERという名称で、開発言語はCOBOLであるということを示している。データ定義情報５０２では画面ファイル２００にnameという名称で定義されたデータが、業務プログラムではUSERNAMEという変数にマッピングし、データ型はkanji型で、データ長は１０文字、埋め字にはスペースを使用するという情報を示している。idという名称で定義されたデータは、業務プログラムではUSERIDという変数にマッピングし、データ型はcharacter型で、データ長は8文字、埋め字にはzero（“0”）を使用するという情報を示している。buttonという名称で定義されたデータは、業務プログラムではSYORIFLUGという変数にマッピングし、データ型はnumber型で、データ長は１文字、埋め字は使用しないという情報を示している。

【００１７】

図６は生成ツール２１０の処理フローを示している。図６の説明は図４、図５、図７、図８、図９を参照しながら行う。図４に示す画面ファイル２００の定義情報２０１を読取る処理６１０で、画面ファイル２００で拡張タグであるjfdで

指定された項目を抜き出し、定義されている属性値から読取った情報で図５に示す定義情報テーブル５００を作成する。定義情報テーブル５００内には、関数情報テーブル５０１とデータ定義情報テーブル５０２がある。

【００１８】

関数情報テーブル５０１の呼出し部名称とは画面に対応した呼出し部の名称を表していて図４の下線部２１２のclassNameの属性値である。パッケージ名称は業務プログラムをdllファイルにした場合のdllのファイル名称を指している。Jfdタグ内のdllNameの属性値である。関数名は呼出す業務プログラムの名称を表していて、programNameの属性値である。言語は業務プログラムの開発言語を指定していて、Wordの属性値である。

【００１９】

データ定義情報テーブル５０２の名称は画面ファイル２００において、実際のフォームオブジェクトに関連付けた名称になっていて図４の下線部２０３のnameの属性値である。COBOL名称は業務プログラム内で、使用されるデータ定義名で、画面から入力されるデータ名を業務プログラム用に別の名前をつけたものである。Jfd:dataタグ内のinterfaceの属性値である。データ型は業務プログラムで使用するデータの型を表していてtypeの属性値である。長さはデータ長を表していてsizeの属性値である。整形はデータに必要となる整形（右寄せ、左寄せなど）を表したもので、alignmentの属性値である。埋め字はデータの余った部分に何かしらの文字を詰める処理のことを指し、そのつめる文字が何であるかを指定することで、paddingCharの属性値である。

【００２０】

業務プログラム雛型２５０の言語設定処理６２０は関数情報テーブル５０１の言語の項目を元に設定される。業務プログラム雛型２５０及びNativeインタフェース（ソース）２４０に呼出す関数名を設定する処理６３０では、関数情報テーブル５０１の関数名の項目が参照される。図９の業務プログラムの雛型２５０のPROGRAM-ID.NEWUSER.が設定される。データ定義情報検出分岐処理６４０では、データ定義情報テーブル５０２から１レコードづつ読み込んでいく。データが定義されている場合はデータ型の項目を参照してデータ型変換メソッドの生成と長

さ、整形及び埋め字の項目を参照してデータ整形メソッド生成の処理６５０を行う。図８に示すプログラム呼出し部（ソース）２３０内のデータ変換部２３１にjfdConvJStrメソッドとjfdLjust,jfdRjustメソッドが生成される。データ取得メソッド生成処理６６０では、画面ファイル２００の中で定義されているオブジェクトのname属性とjfdの拡張タグで定義されているデータのname属性を照らし合わせて、一致した場合は、そこにプログラム呼出し部からデータを取得するためのメソッドを組み込む処理を行う。図７の動的データ参照画面ファイル２２０内の下線部２２２にそれぞれvalue属性値に取得メソッドが追加されている。

【００２１】

次に業務プログラム雛型２５０及びプログラム呼出し部２３０にデータ定義とデータ設定メソッドを設定する処理６７０を行う。図８のデータ変換部２３１及びデータ抽出部２３２にそれぞれのデータ定義に対応して生成される。また図９の業務プログラムの雛型２５０内にもLINKAGE SECTIONのCOPY句の内容として、対応したデータの定義が生成される。終了判定分岐処理６８０で、データ定義が１つもない場合は、エラー処理６８２を行いファイルは何も生成しないで終了する。データ定義が１つ以上ある場合は、それぞれのファイルの生成処理６８１を行い終了する。

【００２２】

図７に動的データ参照画面ファイル２２０の詳細コードを示す。下線部２２１ではプログラム呼出し部にアクセスするための設をしている。classの属性値は画面ファイル２００の下線部２０２のclassNameの属性値である。idの属性値は下線部２０２のidの属性値である。scopeの属性値は下線部２０２のscopeの属性値である。画面ファイル２００のなかでjfd拡張タグで定義されている情報は全て削除する。下線部２２２はフォームオブジェクトの記述部分だが、これは画面ファイル２００内で下線部２０３のデータ定義情報の中に同じnameの属性値がある場合に、そのデータを取得するためのメソッドをvalue属性に指定している。こうすることで、業務アプリケーションから、何らかのデータを取得して初期値として表示することが可能になる。

【００２３】

図８にプログラム呼出し部２３０の詳細コードを示す。データ型の変換、整形処理部２３１では、Nativeインタフェースにデータを設定するsetName, setId, setButtonというメソッドが、画面から入力されるデータに対応して定義されている。それぞれのメソッドは内部でsetUsername, setUserid, setSyoriflugというメソッドを呼出し、実際に業務プログラムにデータを設定する。パラメータとして指定しているデータはjfdConvJStrというメソッドに、文字列化したデータそのものと埋め字用の文字（スペース、ゼロ等）と文字列の長さを表す数字を渡すことで、返り値として指定した長さに、余白部分を指定した埋め文字で埋められた文字列を得る。そのデータをjfdDataRjustまたはjfdDataLjustというメソッドに渡すことで、右寄せ、または左寄せされた文字列データを取得し、そのデータを業務プログラムに設定している。

【００２４】

パラメータ抽出及び設定処理部２３２ではgetParameterメソッドを使ってリクエストの中からname, id, buttonという名称で設定されているパラメータをそれぞれ取得する。また抽出したデータはデータ型の変換、整形処理部２３１で定義された設定メソッドを使用してNativeインタフェースに設定している。

【００２５】

データの抽出とデータの変換、設定処理を全て終わらせてからプログラム呼出し部２３３でcallCOBOLメソッドで業務プログラム呼出し処理を行っている。

【００２６】

図９に業務プログラム雛型２５０の詳細コードを示す。業務プログラム雛型２５０は画面ファイル２００内の定義情報２０１を元に関数名や画面との入出力を行うためのデータ定義が既に記述されている部分と開発者が独自に実装するデータ定義部２５１と実処理部２５２からなる。開発では生成された業務プログラム雛型のファイルに開発者が実処理部２５２をコーディングしてからコンパイルを行い配置することになる。「IDENTIFICATION DIVISION.」、「PROGRAM-ID.」、「DATA DIVISION.」、「WORKING-STORAGE SECTION.」、「LINKAGE SECTION.」、「PROCEDURE DIVISION USING GYOMU-A.」、「EXIT-PROGRAM」および「END-」の部分については、ＣＯＢＯＬ言語のプログラムを生成するためのテンプレ

ートとして記憶している部分であり、業務プログラムの生成時に自動生成する。前記抽出したパラメータから「01　GYOMU-A.」、「02　USERNAME　PIC N(10).」、「02　USERID　PIC X(8).」および「02　SYORIFLUG　PIC X(1).」を生成する。構造体名である「 GYOMU-A」は、システムで自動的に割り付けることも可能であるし、操作者からの入力を受けて設定することも可能である。

【００２７】

図１０にコンパイルされたプログラム呼出し部２７０の詳細図を示す。図３に示す画面３００から名前用のテキストフィールド３１０、パスワード用のテキストフィールド３２０及びボタン３３０が図７に示すようにそれぞれname、id、buttonというパラメータ名でリクエスト１０００内に格納されて送信されてくる。プログラム呼出し部２７０はリクエスト１０００を受け取り、パラメータ抽出部２７１でnameパラメータに格納されているデータ、idパラメータに格納されているデータ、buttonパラメータに格納されたデータをそれぞれ抽出する。業務プログラムに適した型や、形に変換するデータ変換部２７３で抽出されたデータは図４の画面ファイルに示されているように、nameは業務プログラムのUSERNAMEという変数にマッピングし、kanji型でデータ長は１０文字、埋め字はスペースとなるように変換される。idは業務プログラムのUSERIDという変数にマッピングし、character型でデータ長は８文字、右寄せで、埋め字は”０”となるように変換される。buttonは業務プログラムのSYORIFLUGという変数にマッピングし、number型でデータ長は１文字となるように変換される。

【００２８】

プログラム呼出し部２７２は図８に示すように抽出したデータをデータ変換し、その結果業務プログラムに必要なパラメータ１０１０をNativeインタフェースに渡す。更に必要なパラメータを全てを設定した後に業務プログラムを実行させる。

【００２９】

図１１にデータ変換部の処理フローを示す。言語分岐１１００、Java用型変換処理１１１０、Java用けた寄せ処理１１１１、Java用埋め字処理１１１２、COBOL用型変換処理１１２０、COBOL用けた寄せ処理１１２１、COBOL用埋め字処理１

１２２、C言語用型変換処理１１３０、C言語用けた寄せ処理１１３１、C言語用埋め字処理１１３２からなる。

図１２にけた寄せと埋め字について示す。記号は説明１２００を参照する。何もしない場合は１２１０の入力に対して、１２１１のマッピングになる。右にけた寄せを行うと１２２０の入力に対して１２２１のように格納される。左にけた寄せを行い、埋め字スペースを用いると、１２３０は１２３１のように格納される。右寄せ、埋め字がスペースだと１２４０は１２４１のように格納される。数値に対し、何も行わないと１２５０は１２５１のように格納される。数値に対し、右寄せを行い埋め字半角スペースを用いると１２６０は１２６１のように格納される。右寄せ、０を埋め字に用いると１２７０は１２７１のように格納される。

【００３０】

図１３に本発明を用いた開発環境のエディタ１３００の概念図を示す。画面を実際に表示させる画面ビュー１３１０を見ながら、画面ファイルをコードビュー１３２０から編集を行う。また、画面ファイルの定義情報から生成される業務プログラムの雛型が業務プログラムビュー１３３０に表示されるので、開発者は、開発者が実装する処理部１３３１を編集し、業務プログラムを作成する。

生成物の全体の流れを説明する。

【００３１】

動的データ参照画面ファイル２２０で表示される画面３００のテキストフィールド３１０に名前を入力しテキストフィールド３２０にパスワードを入力したあと送信ボタン３３０を押下すると名前はname、パスワードはid、送信ボタンを押下した情報はbuttonというパラメータとして設定され、パラメータはその他の送信に必要なデータとともに連結されて、リクエスト１０００という形になり図１に示すネットワーク１４０を介してWebサーバ１５０上にあるServlet１６０に送信される。Servlet１６０ではリクエスト１０００を受け取ると画面に対応したプログラム呼出し部にリクエスト１０００をそのまま転送する。

【００３２】

プログラム呼出し部２７２はパラメータ抽出部２７１でリクエスト１０００を受け取るとgetParameterメソッドで、name、id、button、というパラメータをそ

れぞれ抽出し、データ変換部２７３でそれぞれのデータをnameはkanji型、１０文字、スペースで埋め字処理、idはcharacter型で８文字、右寄せで格納し”０”で埋め字処理、buttonはnumber型で１文字というように変換を行う。それぞれのパラメータは業務プログラムでnameはUSERNAME、idはUSERID、buttonはSYORIFLUGと関連づいているので、Nativeインタフェース（COBOL）１７０の設定メソッドを使用して、パラメータを構築する。パラメータの構築が終わり次第、プログラム呼出し部２７２がCOBOLの業務プログラムを実行する。図９の業務プログラム雛型２５０内の開発者実装部分２５１及び２５２をコーディングしてコンパイルした、業務プログラム２９０は実行すると、あらかじめNativeインタフェース２８０が業務プログラム２９０からパラメータが参照できるように設定しているので、構築されたパラメータを参照しUSERNAME、USERID、SYORIFLUGを使用した処理を行う。

【００３３】

【発明の効果】

本発明によれば、種々のプログラム言語でコーディングされた業務プログラムを容易に連携することが可能となる。

【図面の簡単な説明】

【図１】本発明の生成物の構成の実施例である。

【図２】本発明のプログラム生成の概念図である。

【図３】本発明の画面ファイルの表示例である。

【図４】本発明の画面ファイルの詳細コードである。

【図５】本発明の定義情報のテーブルである。

【図６】本発明の生成ツールの処理フローである。

【図７】本発明の動的データ参照画面ファイルの詳細コードである。

【図８】本発明のプログラム呼出し部の詳細コードである。

【図９】本発明の業務プログラム雛型の詳細コードである。

【図１０】本発明のプログラム呼出し部の概念図である。

【図１１】本発明のデータ変換の処理フローである。

【図１２】本発明のデータ変換の実施例である。

【図１３】本発明の開発環境エディタの概念図である。

【符号の説明】

１００ 端末

１１０ 画面Ａ

１１１ プログラム呼出し部Ａ

１１２ 業務プログラムＡ（COBOL）

１２０ 画面Ｂ

１２１ プログラム呼出し部Ｂ

１２２ 業務プログラムＢ（COBOL）

１３０ 画面Ｃ

１３１ プログラム呼出し部Ｃ

１３２ 業務プログラムＡ（Java）

１４０ ネットワーク

１５０ Webサーバ

１６０ Servlet

１７０ Nativeインタフェース（COBOL）

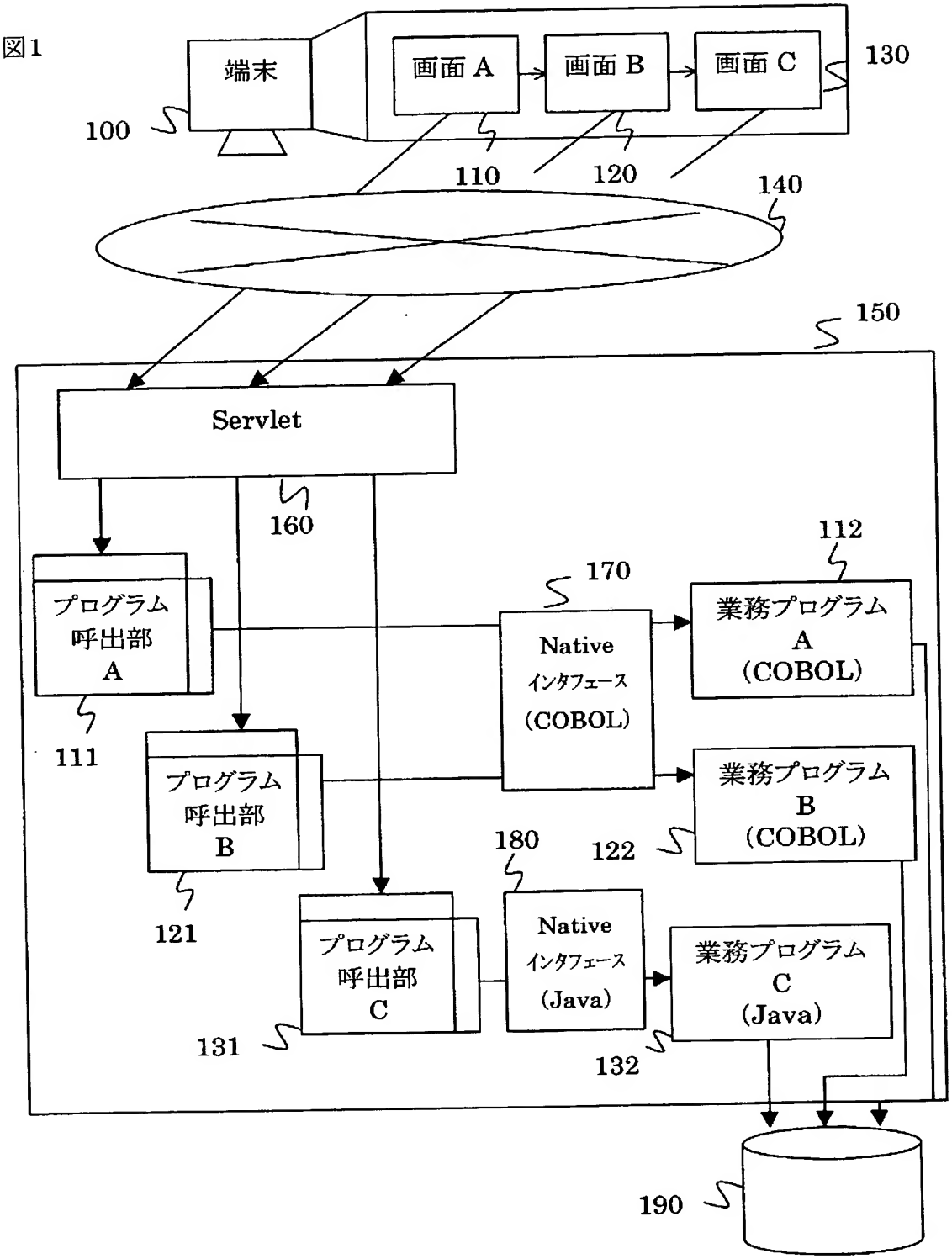
１８０ Nativeインタフェース（Java）

１９０ データベース

【書類名】　図面

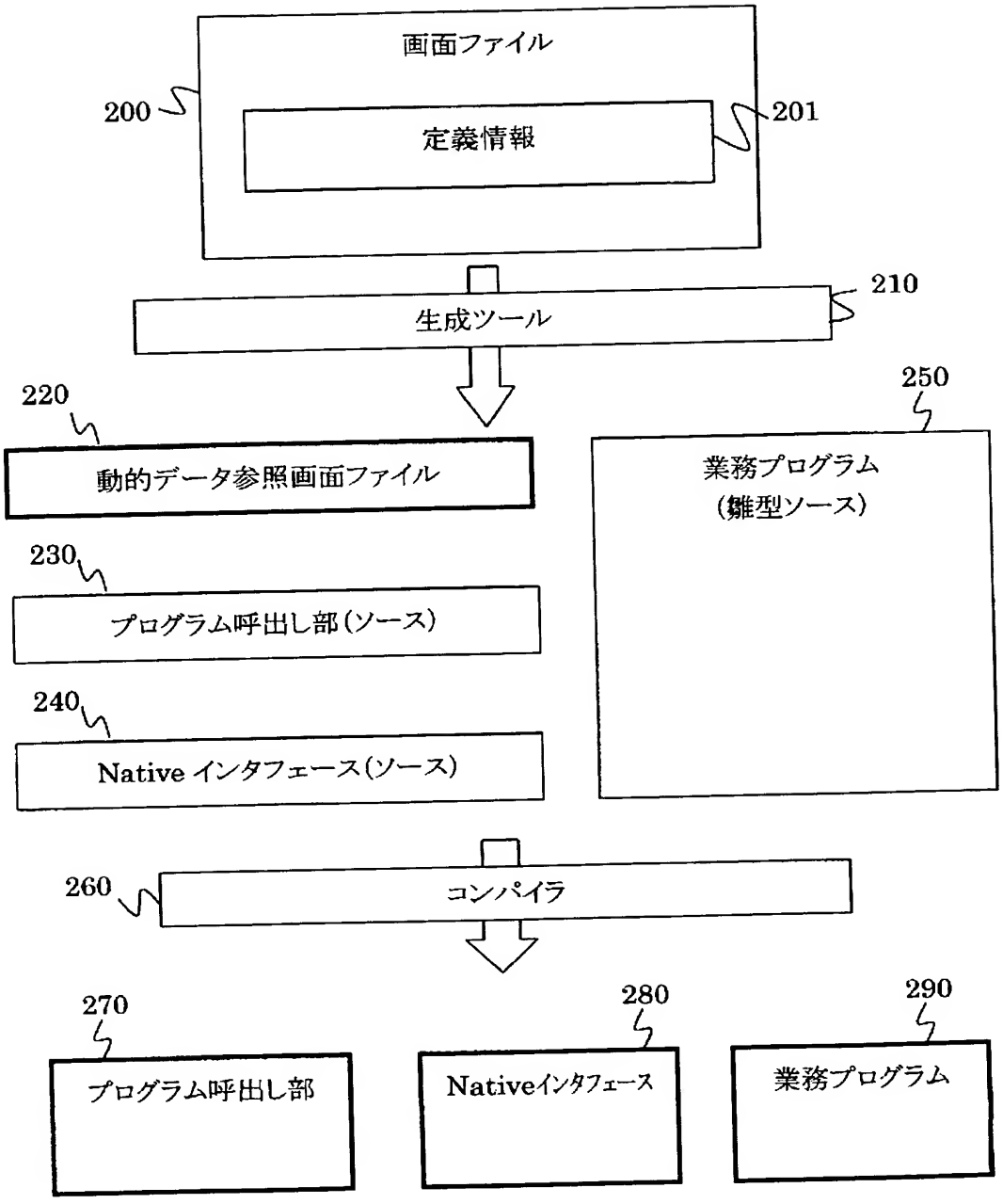
【図１】

図1


端末
100
画面 A
画面 B
画面 C
130
110
120
140
150
Servlet
160
プログラム
呼出部
A
111
プログラム
呼出部
B
121
プログラム
呼出部
C
131
170
Native
インタフェース
(COBOL)
112
業務プログラム
A
(COBOL)
業務プログラム
B
(COBOL)
122
180
Native
インタフェース
(Java)
業務プログラム
C
(Java)
132
190

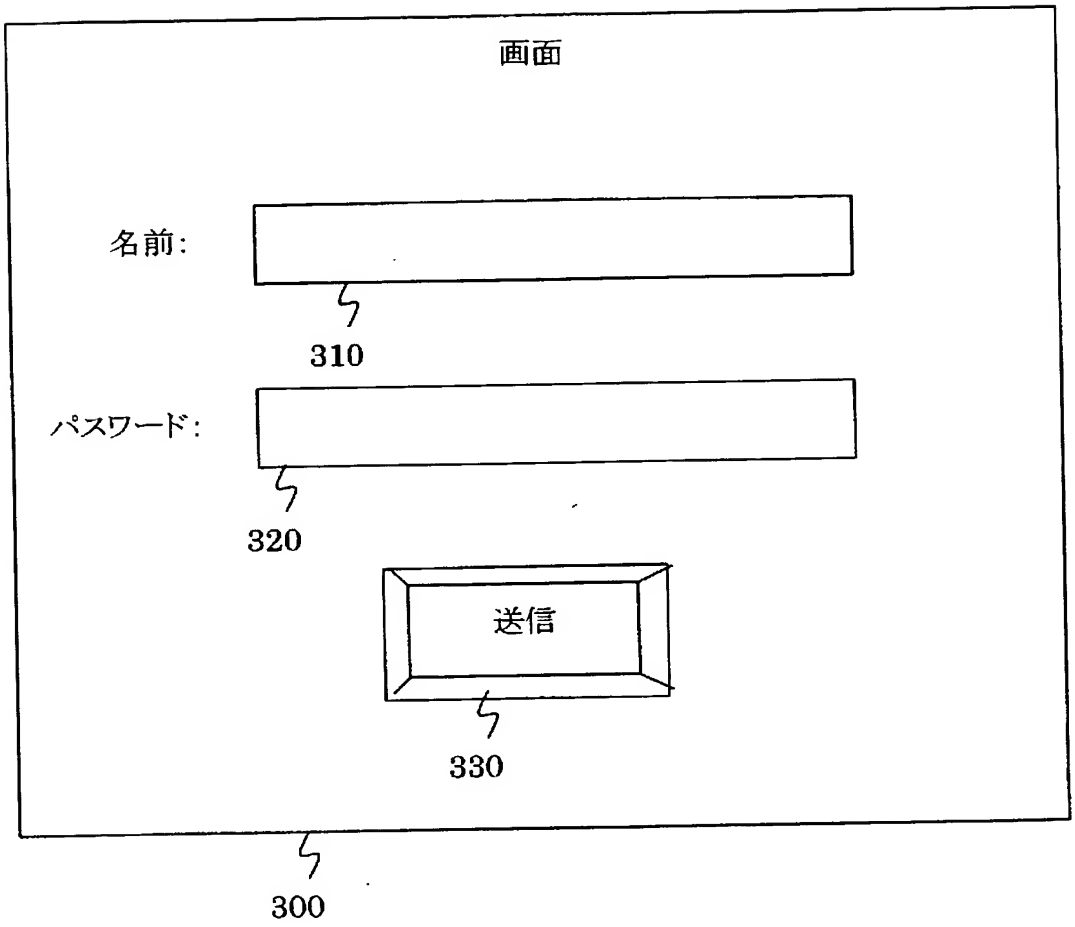
【図２】

図2


画面ファイル
200
201
定義情報
210
生成ツール
220
動的データ参照画面ファイル
250
業務プログラム
(雛型ソース)
230
プログラム呼出し部(ソース)
240
Native インタフェース(ソース)
260
コンパイラ
270
プログラム呼出し部
280
Nativeインタフェース
290
業務プログラム

【図３】

図3



【図４】

図4

画面ファイル

```
<html>
<head>
<title>サンプル</title>
<meta http-equiv="Content-Type" content="text/html; charset=Shift_JIS">
<jfd:dataDefinition scope="session" className="Sample.Page"
    dllName="SAMPLE" id="SamplePage" programName="NEWUSER"
    Word="COBOL">
<jfd:data name="name" interface="USERNAME"size="10"
            type="kanji" paddingChar="space" />
<jfd:data name="id" interface="USERID" size="8"
            type="character" alignment="right" paddingChar="zero" />
<jfd:data name="button" interface="SYORIFLUG" size="1"
            type="number" />
</jfd:dataDefinition>
</head>
<body bgcolor="#FFFFFF" text="#000000">
<form name="form1" method="post" action="">
    <input type="text" name="name">
    <input type="text" name="id">
    <input type="submit" name="button" value="送信">
</form>
</body>
</html>
```

202

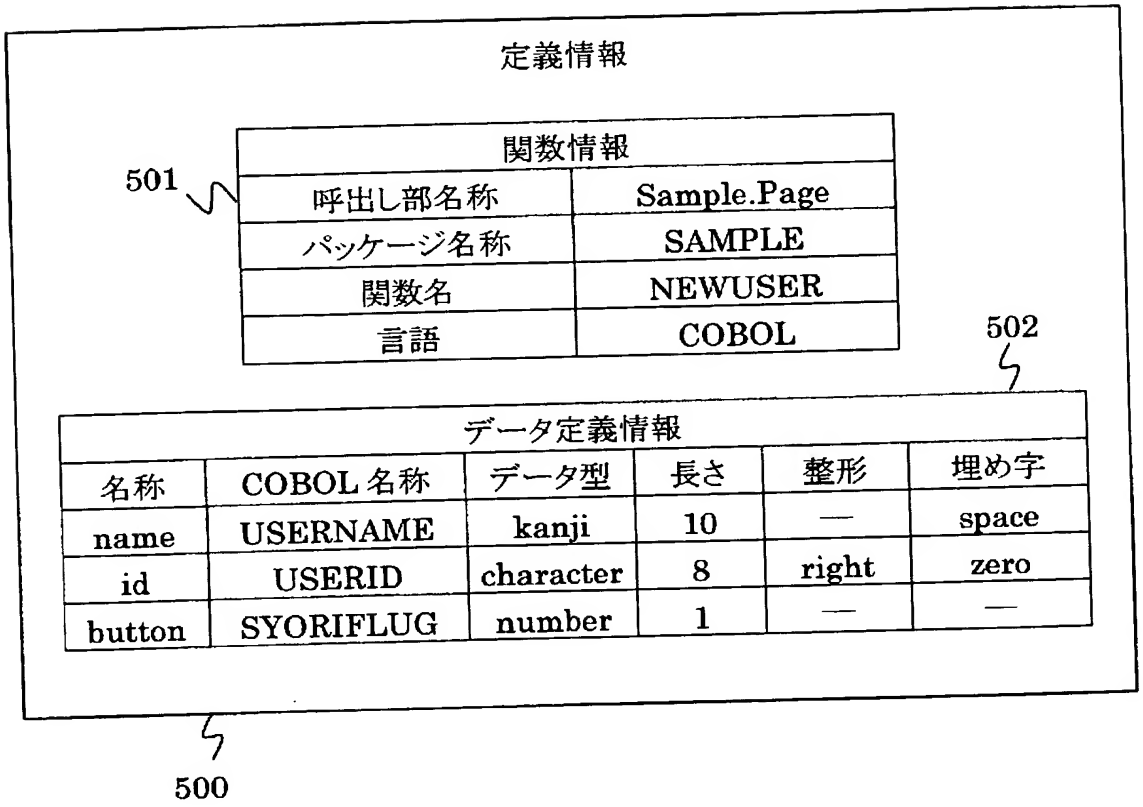
203

201

200

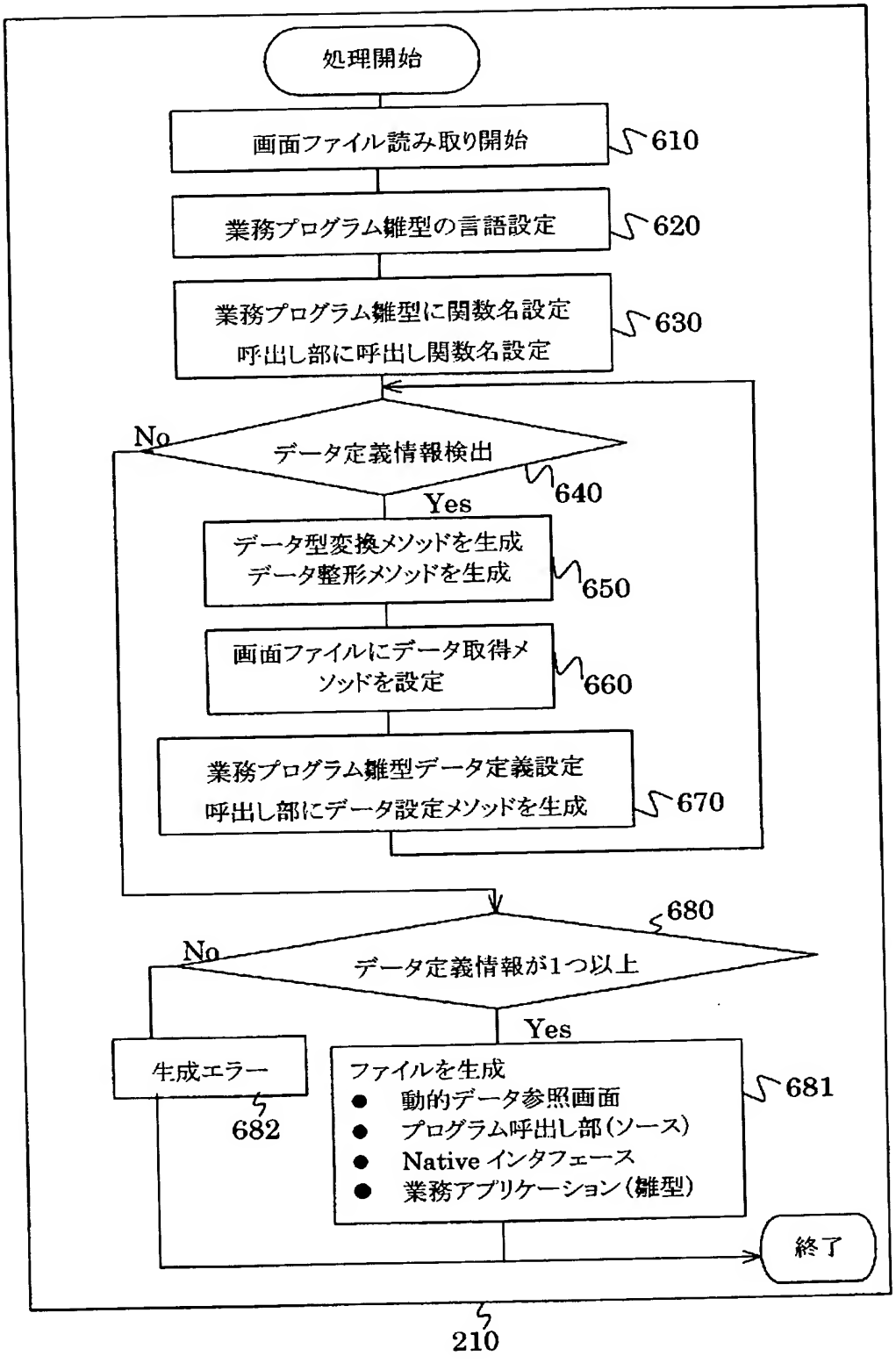
【図５】

図5


定義情報
501
関数情報
呼出し部名称
Sample.Page
パッケージ名称
SAMPLE
関数名
NEWUSER
言語
COBOL
502
データ定義情報
名称
COBOL名称
データ型
長さ
整形
埋め字
name
USERNAME
kanji
10
—
space
id
USERID
character
8
right
zero
button
SYORIFLUG
number
1
—
—
500

【図６】

図6


処理開始
画面ファイル読み取り開始
610
業務プログラム雛型の言語設定
620
業務プログラム雛型に関数名設定
呼出し部に呼出し関数名設定
630
No
データ定義情報検出
640
Yes
データ型変換メソッドを生成
データ整形メソッドを生成
650
画面ファイルにデータ取得メソッドを設定
660
業務プログラム雛型データ定義設定
呼出し部にデータ設定メソッドを生成
670
680
No
データ定義情報が1つ以上
Yes
生成エラー
682
ファイルを生成
● 動的データ参照画面
● プログラム呼出し部(ソース)
● Native インタフェース
● 業務アプリケーション(雛型)
681
終了
210

【図７】

図7

動的データ参照画面ファイル

```
<%@ page contentType="text/html; charset=Shift_JIS" %>
<%@ page import="java.io.File" %>
<jsp:useBean class="Sample.Page" id="SamplePage" scope="session" />
<html>
<head>
<title>サンプル</title>
<meta http-equiv="Content-Type" content="text/html; charset=Shift_JIS">
</head>

<body bgcolor="#FFFFFF" text="#000000">

<form name="form1" method="post" action="Sample">
    <input type="text" name="name" value="<%= SamplePage.getName_value() %>">
    <input type="text" name="id" value="<%= SamplePage.getId_value() %>">
    <input type="submit" name="button"
           value="<%= SamplePage.getButton_value() %>">
</form>

</body>
</html>
```

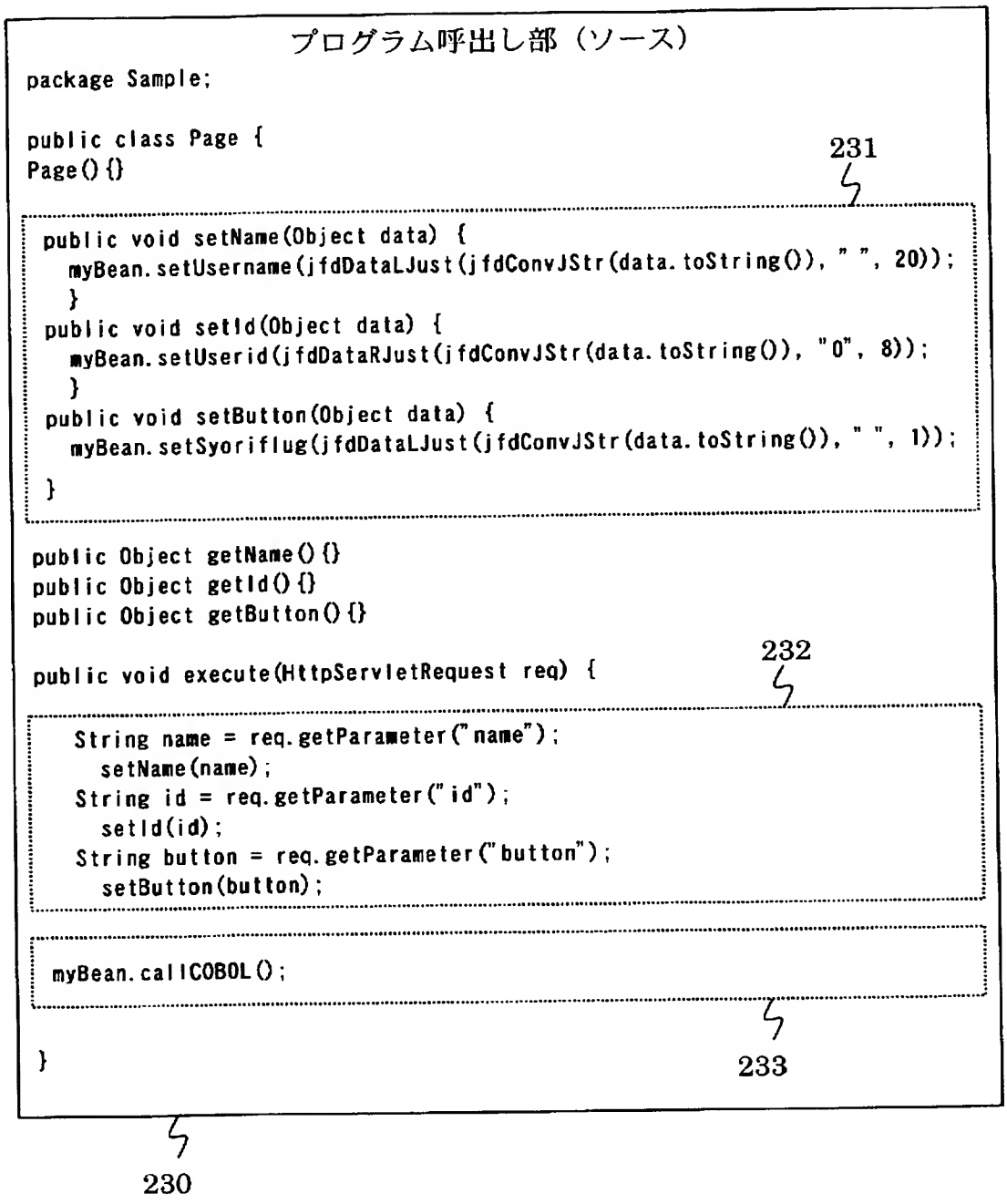
221

222

220

【図８】

図8


プログラム呼出し部（ソース）
package Sample;
public class Page {
Page() {}
231
public void setName(Object data) {
myBean.setUsername(jfdDataLJust(jfdConvJStr(data.toString()), " ", 20));
}
public void setId(Object data) {
myBean.setUserid(jfdDataRJust(jfdConvJStr(data.toString()), "0", 8));
}
public void setButton(Object data) {
myBean.setSyoriflug(jfdDataLJust(jfdConvJStr(data.toString()), " ", 1));
}
public Object getName() {}
public Object getId() {}
public Object getButton() {}
public void execute(HttpServletRequest req) {
232
String name = req.getParameter("name");
setName(name);
String id = req.getParameter("id");
setId(id);
String button = req.getParameter("button");
setButton(button);
myBean.callCOBOL();
233
}
230

【図９】

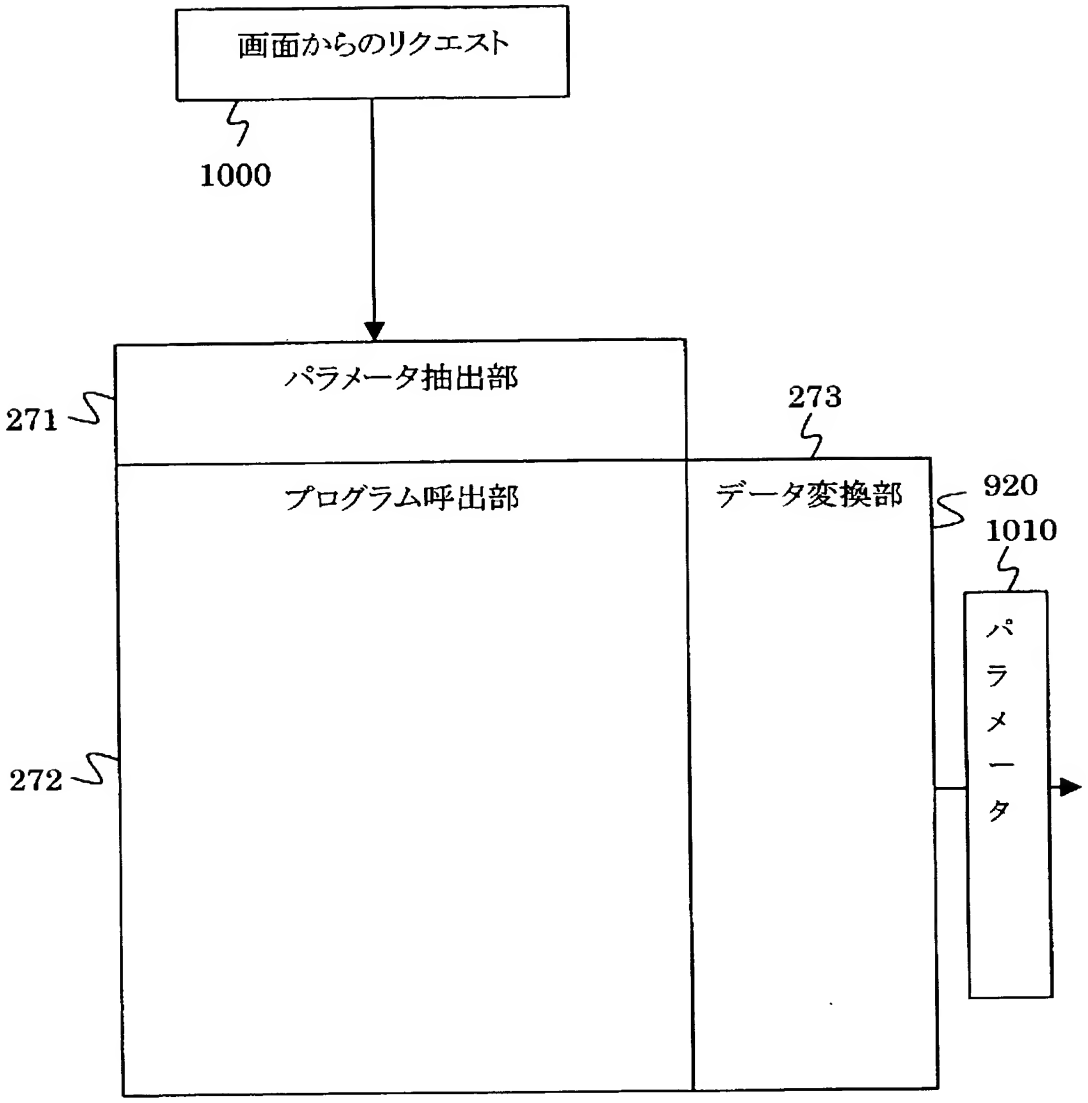
図９

```
                業務プログラム（COBOL）
IDENTIFICATION  DIVISION.
PROGRAM-ID.  NEWUSER.

DATA  DIVISION.
WORKING-STORAGE  SECTION.                          251
                                                    ↓
 01  SYSTEMNAME                    PIC  N(10).
 01  SYSTEMID                      PIC  X(8).

LINKAGE  SECTION.

  01  GYOMU-A.
    02  USERNAME                   PIC N(10).
    02  USERID                      PIC X(8).
    02  SYORIFLUG                  PIC X(1).

PROCEDURE  DIVISION  USING  GYOMU-A.          252
                                                    ↓
   MOVE  USERNAME  TO  SYSTEMNAME
   MOVE  USERID    TO  SYSTEMID
   MOVE     0      TO  SYORIFLUG

EXIT-PROGRAM

END-NEWUSER.
```

250

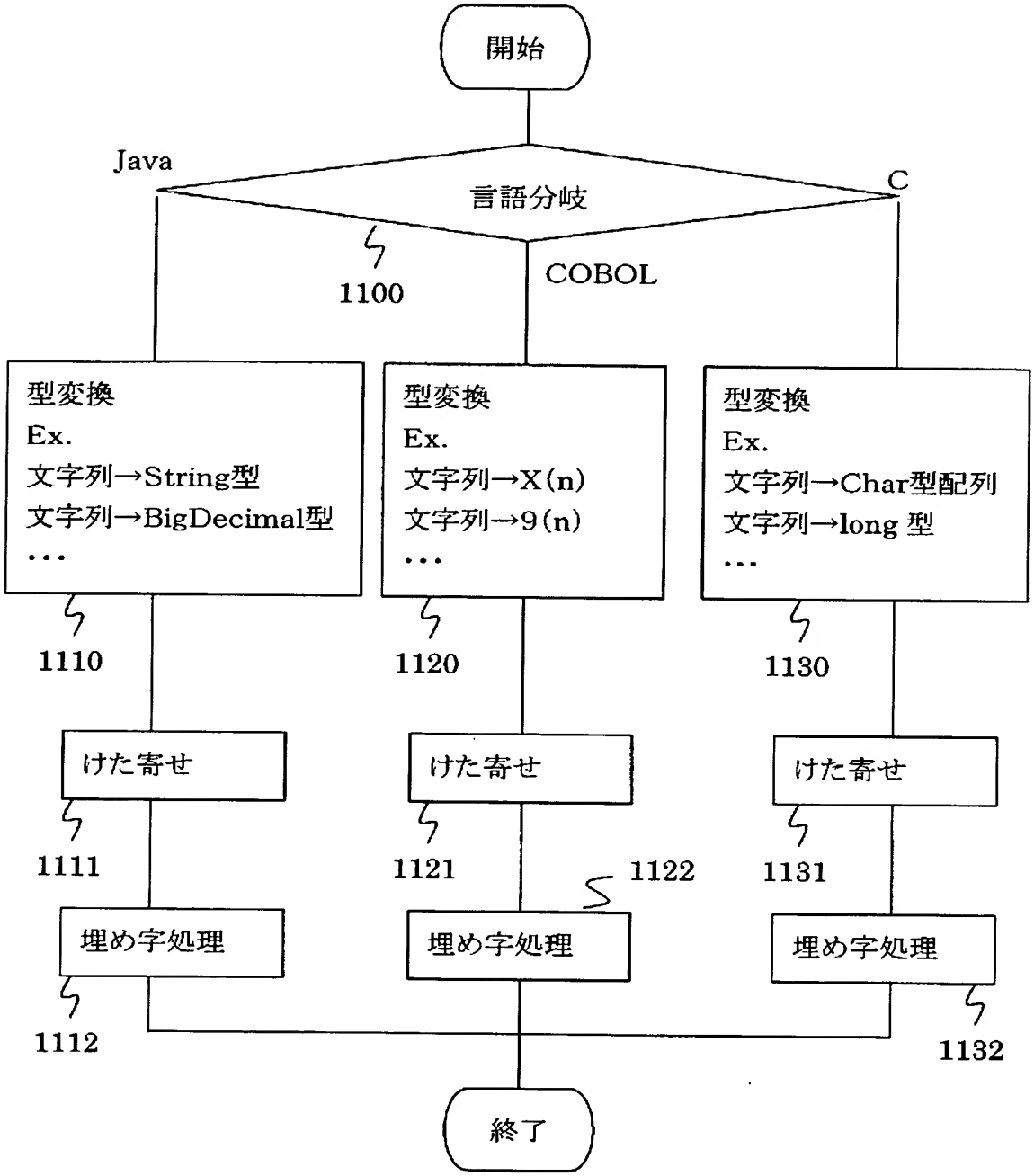
【図１０】

図10


画面からのリクエスト
1000
パラメータ抽出部
271
273
プログラム呼出部
データ変換部
920
1010
パラメータ
272

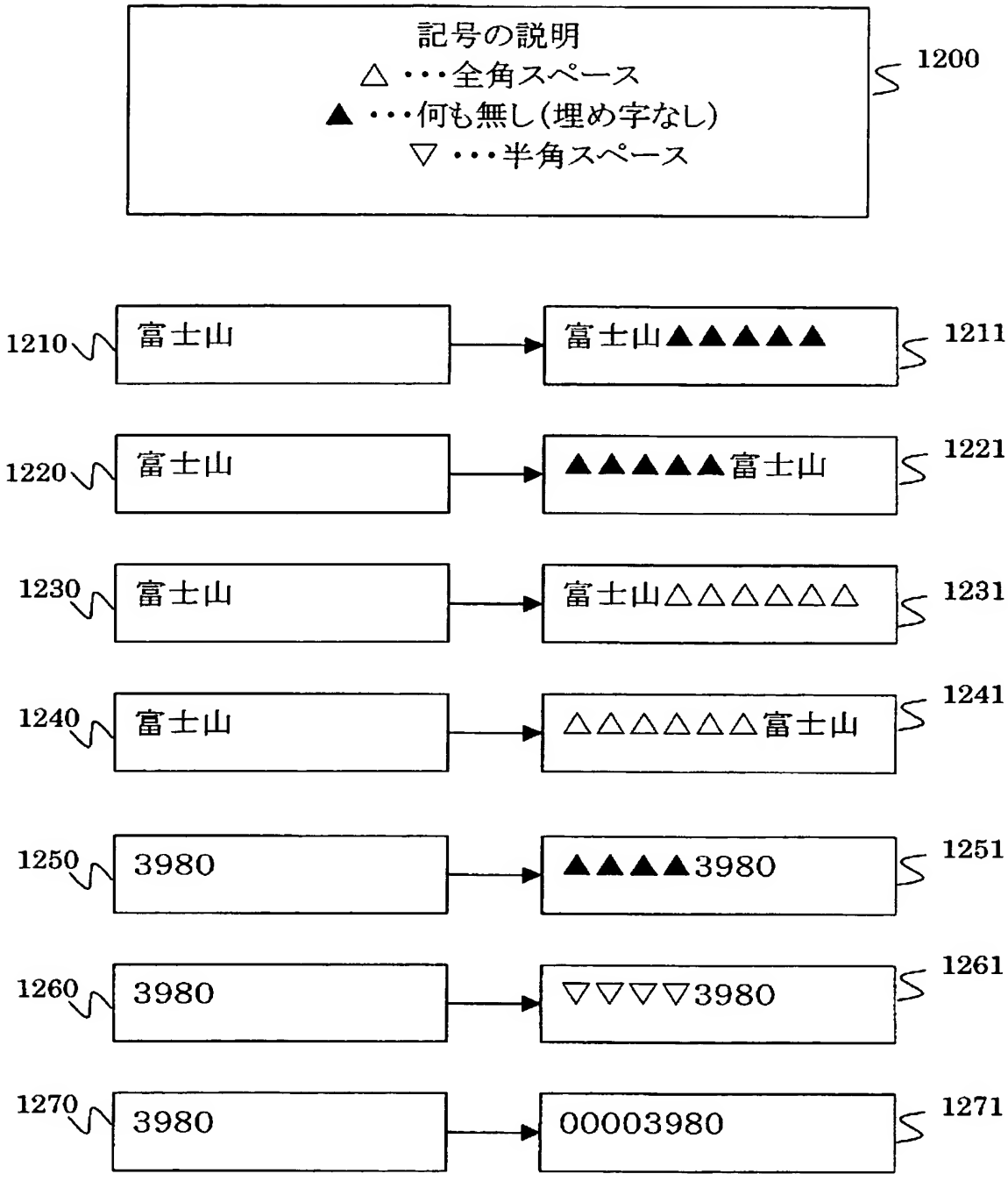
【図１１】

図11


開始
Java
言語分岐
C
COBOL
1100
型変換
Ex.
文字列→String型
文字列→BigDecimal型
・・・
型変換
Ex.
文字列→X(n)
文字列→9(n)
・・・
型変換
Ex.
文字列→Char型配列
文字列→long 型
・・・
1110
1120
1130
けた寄せ
けた寄せ
けた寄せ
1111
1121
1122
1131
埋め字処理
埋め字処理
埋め字処理
1112
1132
終了

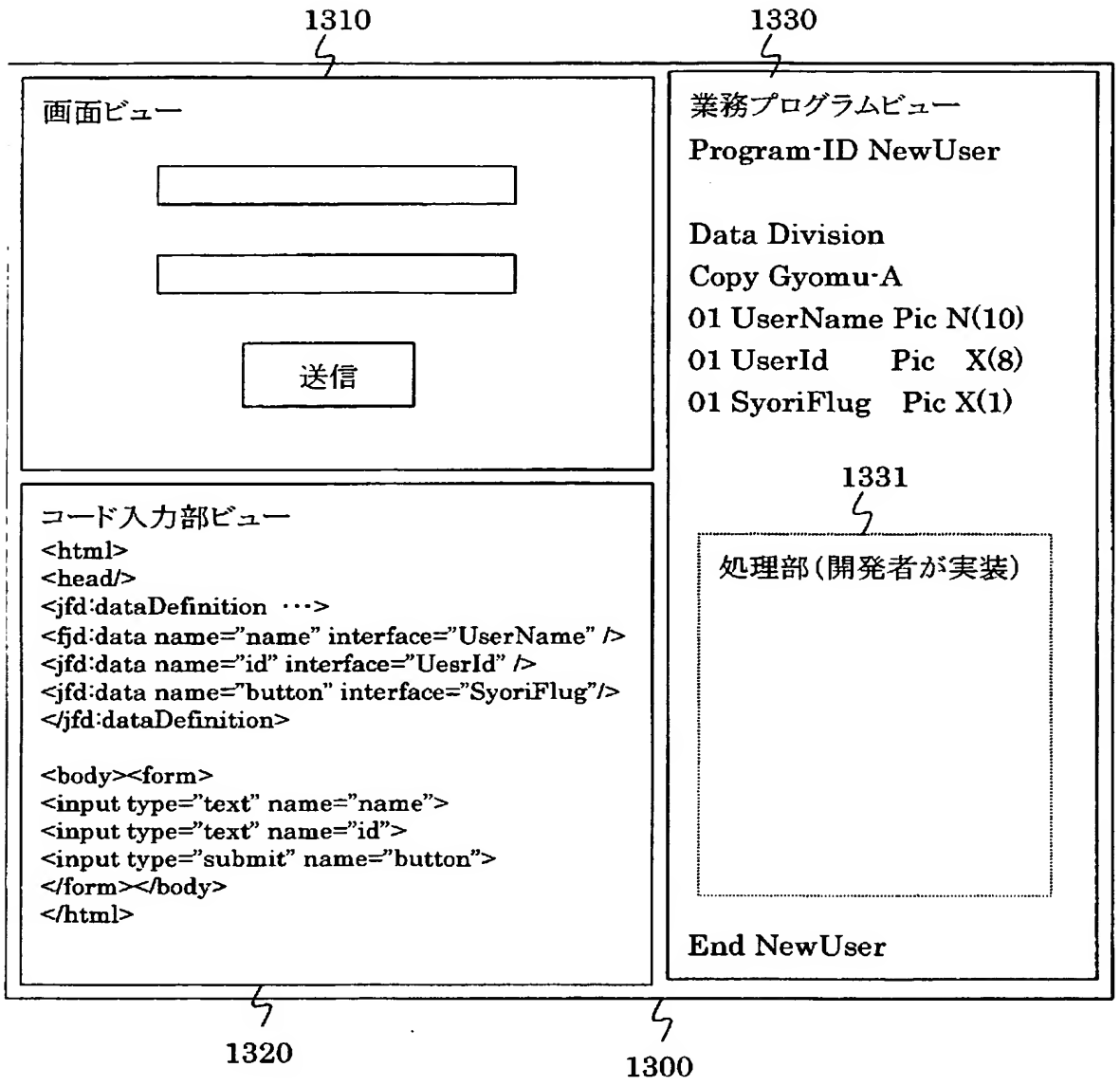
【図１２】

図12


記号の説明
△・・・全角スペース
▲・・・何も無し(埋め字なし)
▽・・・半角スペース
1200
1210
富士山
富士山▲▲▲▲▲
1211
1220
富士山
▲▲▲▲▲富士山
1221
1230
富士山
富士山△△△△△△
1231
1240
富士山
△△△△△△富士山
1241
1250
3980
▲▲▲▲3980
1251
1260
3980
▽▽▽▽3980
1261
1270
3980
00003980
1271

【図１３】

図13


1310
1330
画面ビュー
送信
業務プログラムビュー
Program-ID NewUser
Data Division
Copy Gyomu-A
01 UserName Pic N(10)
01 UserId Pic X(8)
01 SyoriFlug Pic X(1)
1331
処理部(開発者が実装)
End NewUser
コード入力部ビュー
<html>
<head/>
<jfd:dataDefinition ···>
<fjd:data name="name" interface="UserName" />
<jfd:data name="id" interface="UesrId" />
<jfd:data name="button" interface="SyoriFlug"/>
</jfd:dataDefinition>
<body><form>
<input type="text" name="name">
<input type="text" name="id">
<input type="submit" name="button">
</form></body>
</html>
1320
1300

【書類名】　要約書

【要約】

【課題】画面ファイルからの情報を用いて、バックエンドのCOBOLのような言語でコーディングされたプログラムと連携する為の、業務プログラムの雛型を作成する。

【解決手段】画面ファイルに必要となるフォームの入力データ定義を行い、またバックエンドの業務プログラムの関数名、開発言語を定義させることで、型変換、パラメータの設定を隠蔽する装置を生成させ、それを実装する業務プログラムの雛型を生成する。

【選択図】　図１

認定・付加情報

特許出願の番号	特願２００２－１８８９３８
受付番号	５０２００９４７３４１
書類名	特許願
担当官	第七担当上席　　００９６
作成日	平成１４年　７月　１日

＜認定情報・付加情報＞

【提出日】　平成14年 6月28日

次頁無

特願２００２－１８８９３８

出　願　人　履　歴　情　報

識別番号　　［０００００５１０８］

１．変更年月日　　１９９０年　８月３１日
［変更理由］　　新規登録
住　所　　東京都千代田区神田駿河台４丁目６番地
氏　名　　株式会社日立製作所